Update on the Development and Validation of MERCURY: A Modern, Monte Carlo Particle Transport Code

R. J. Procassini, J. M. Taylor, M. S. McKinley, G. M. Greenman, D. E. Cullen, M. J. O'Brien, B. R. Beck, C. A. Hagmann

June 6, 2005

International Topical Meeting on Mathematics and Computations
Avignon, France
September 12, 2005 through September 15, 2005

Mathematics and Computation, Supercomputing, Reactor Physics and Nuclear and Biological Applications
Palais des Papes, Avignon, France, September 12-15, 2005, on CD-ROM, American Nuclear Society, LaGrange Park, IL (2005)

UPDATE ON THE DEVELOPMENT AND VALIDATION OF MERCURY: A MODERN, MONTE CARLO PARTICLE TRANSPORT CODE

Richard Procassini, Janine Taylor, Scott McKinley, Gregory Greenman, Dermott Cullen, Matthew O'Brien, Bret Beck and Christian Hagmann
Lawrence Livermore National Laboratory
Mail Stop L-95, P. O. Box 808
Livermore, CA 94551
United States of America
spike@llnl.gov; taylorj@llnl.gov; quath@llnl.gov; greenman@llnl.gov;
cullen1@llnl.gov; mobrien@llnl.gov; bbeck@llnl.gov; hagmann1@llnl.gov

ABSTRACT

An update on the development and validation of the MERCURY Monte Carlo particle transport code is presented. MERCURY is a modern, parallel, general-purpose Monte Carlo code being developed at the Lawrence Livermore National Laboratory. During the past year, several major algorithm enhancements have been completed. These include the addition of particle trackers for 3-D combinatorial geometry (CG), 1-D radial meshes, 2-D quadrilateral unstructured meshes, as well as a feature known as *templates* for defining recursive, repeated structures in CG. New physics capabilities include an elastic-scattering neutron thermalization model, support for continuous energy cross sections and $S(\alpha,\beta)$ molecular bound scattering. Each of these new physics features has been validated through code-to-code comparisons with another Monte Carlo transport code. Several important computer science features have been developed, including an extensible input-parameter parser based upon the XML data description language, and a dynamic load-balance methodology for efficient parallel calculations. This paper discusses the recent work in each of these areas, and describes a plan for future extensions that are required to meet the needs of our ever expanding user base.

KEYWORDS: Monte Carlo, particle transport, parallel computation, code validation

1. INTRODUCTION

Recent advances in the development and validation of the MERCURY Monte Carlo particle transport code [1]are described in this paper. MERCURY is a modern code that is being developed at the Lawrence Livermore National Laboratory (LLNL). The intent is that MERCURY will eventually replace the aging legacy codes TART [2] and COG [3] as the next-generation, general-purpose radiation transport code at LLNL. The code is being developed with both programmatic and institutional funding, by a code team that is drawn from multiple organizations at the laboratory. This approach will ensure that MERCURY will be applicable for solving the broad range of transport problems that are encountered at LLNL.

The breadth of MERCURY's capabilities has steadily increased over the past two years [4]. The code is now able to model the time-dependent transport of multiple particle species though various problem geometries. The particle types that are currently supported include neutrons (n), gammas (γ) and the five low-mass ions $({}^1H, {}^2H, {}^3H, {}^3He, {}^4He)$. The code supports wide variety of geometries, including 1-D radial meshes, 2-D $r - z$ structured and quadrilateral unstructured meshes, 3-D Cartesian structured and tetrahedral unstructured meshes, and 3-D combinatorial geometries (CG). Multigroup and continuous energy (CE) representations of the nuclear data are now supported. A realistic neutron thermalization treatment has been added, and $S(\alpha, \beta)$ bound molecular is currently being tested. Integrated energy deposition, mass production/depletion, and thermally-broadened cross section data are also undergoing testing at the current time. The code has been enhanced through the addition of a dynamic load-balancing capability to improve the efficiency of parallel calculations. Finally, a new input parameter parser has been developed, using the XML data description language, which permits easy modification and extension of the input syntax in the future.

The organization of this paper as follows. Recent algorithm enhancements are discussed in Section 2, while new physics capabilities are presented in Section 3. Recent computer science enhancements are described in Section 4. Finally, the summary of this recent activity and the plans for future work are presented in Section 5.

2. RECENT ALGORITHM ENHANCEMENTS

The majority of the algorithm enhancements made to MERCURY during the past two years have been focused in the area of particle trackers for additional problem geometries. Prior to conversion of the code for general-purpose applications, all supported geometries were meshed based: 2-D $r - z$ structured, and 3-D Cartesian structured and tetrahedral unstructured. Since that time, trackers for 3-D combinatorial geometry (CG), and 1-D radial and 2-D quadrilateral unstructured meshes have been added. In addition, a feature known as *templates* has been developed that permits easy definition of repeated structures in CG.

2.1 3-D Combinatorial Geometry Particle Tracker

The development of a particle tracker for 3-D CG was crucial if the code was to be used for general purpose applications. The first phase of the new CG tracker supports cells defined from the logical aggregation of first-order (planar) and second-order (spherical, elliptical, cylindrical and conical) analytic surfaces. During this first phase, the only supported logical aggregation operation is an *implicit* Boolean `AND' operation. While it is possible to generate very complex CG systems using only `AND' operators, it is not always an efficient process. Therefore, the second phase of the CG capability will support a broad range of logical operations, including Boolean `OR' and `NOT'. This feature has been implemented, but has not been fully tested at this writing. The third phase of the CG capability will include support for fourth-order (toroidal) analytic surfaces, spline-based surfaces of revolution, multidimensional spline surfaces and topographic surfaces.

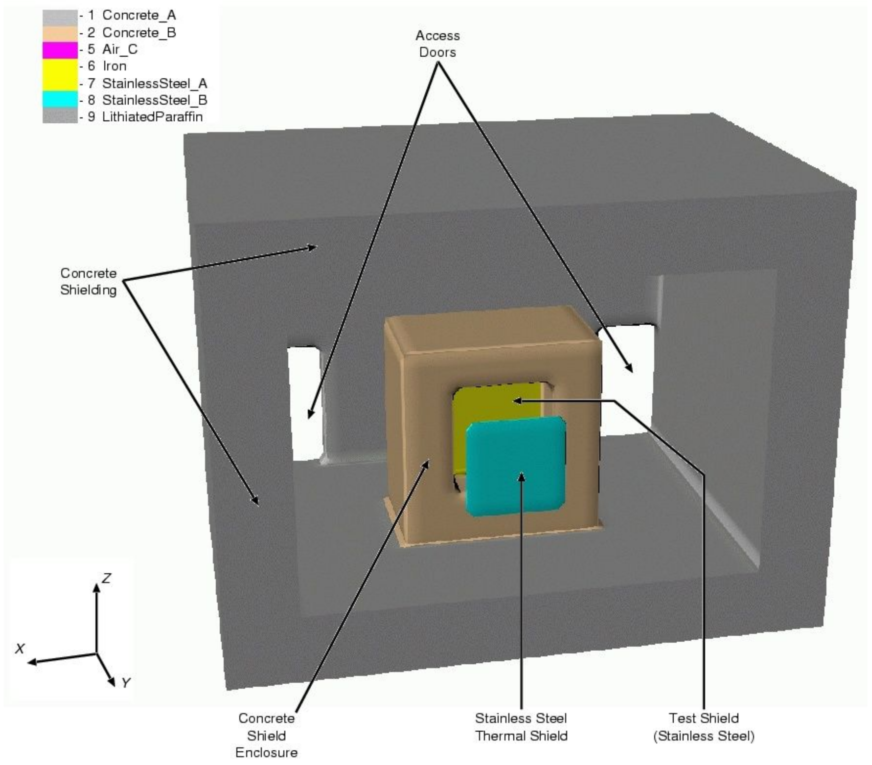


Figure 1. The layout of the fusion-neutron shielding test facility.

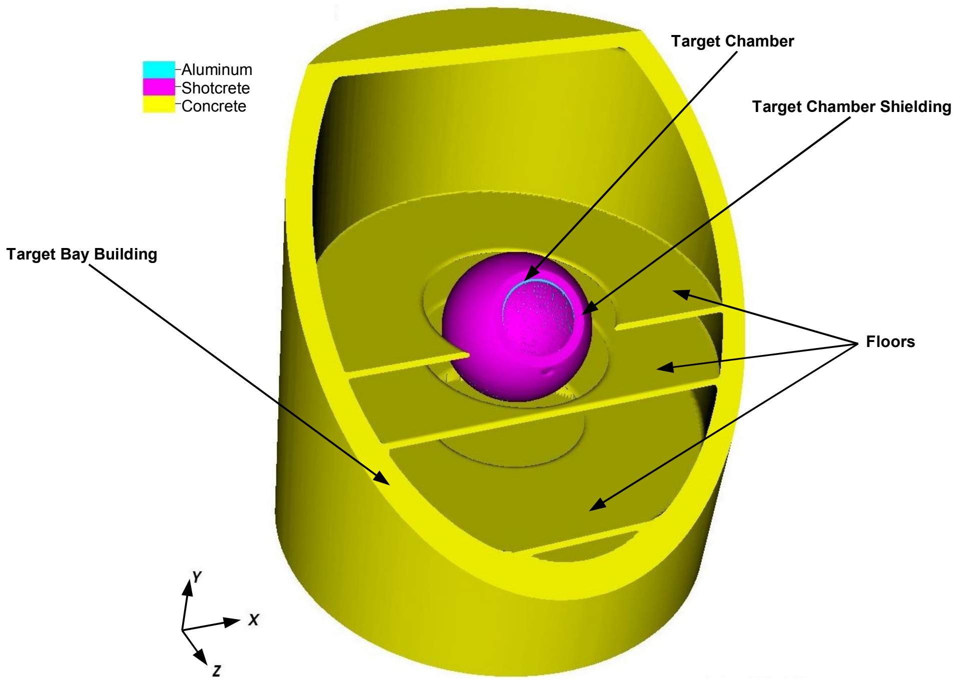


Figure 2. A cutaway view of the target chamber in the target-bay building of the National Ignition Facility (NIF) laser fusion facility. The aluminum target chamber is shown in blue, the shotcrete shielding is purple and the concrete structure is gold.

Two systems that have recently been modeled using the CG tracker are shown in Figures 1 and 2. While neither of these systems are very complex, they are representative of what users of the code are currently modeling. The fusion shielding experiment shown in Figure 1 was a facility at the Oak Ridge National Laboratory (ORNL) that was used to test the effectiveness of various shielding configurations for use in potential fusion reactors. The internals of the room have been exposed by removing the front wall. The (gold) shield is shown within the (bronze) concrete shield enclosure, which sits in front of the 14 MeV neutron beam line. The (turquoise) thermal shield is obscuring the spherical detector. The (gray) structural concrete walls provide shielding for workers and other structures at the site.

Figure 2 shows a cutaway view of the spherical target chamber sitting inside the target-bay building at the National Ignition Facility (NIF). NIF, the largest laser fusion experimental facility in the world, is located at the Lawrence Livermore National Laboratory (LLNL). The target chamber is a composite system composed of an aluminum spherical chamber (blue) that is encased in a spherical layer of shotcrete shielding (purple). The cylindrical walls and floors of the target-bay building are made of concrete (gold) that acts as a second radiation shield. Note that this cutaway view does not show the inside of the target chamber as (blue) aluminum, which it should. The reason for this is the limited resolution of the "graphics mesh" that was superimposed on the CG in order to produce this image. There are plans to remove this limitation in the near future, as will be discussed in Section 6.

2.2 New Mesh-Based Particle Trackers

Two additional mesh-based particle trackers have been developed over the past year. The first is a 1-D radial mesh tracker for modeling spherical systems. This capability uses components of the 3-D CG tracker described above. While this type of system can be described as 1-D oriented along the radius axis of a sphere, it is actually modeled as concentric, nested 3-D spheres which are defined using the same second-order spherical surfaces that are used in 3-D CG systems. The other new mesh based tracker supports 2-D quadrilateral meshes, where the edges of the cells can be aligned at arbitrary angles to either of the (r, z) axes. Since this type of mesh is axisymmetric about the z axis, these cell edges are actually second-order conical surfaces that may be degenerate in the form of cylinders or planes.

2.3 Complex Geometry Generation via Templates

Templates are a recent addition to the code that simplify the generation of complex combinatorial geometries. This capability provides a straightforward, recursive mechanism for defining hierarchical, repeated structures. Simple geometric structures can be defined, and then *referred to* in the definition of other, more-complex geometric structures. The intent of this process is to minimize the number of geometric structures that need to be defined in order to create complex systems. Once all of the required templates have been defined, the user *instantiates* the actual cells through a small number of creation commands.

The essence of the method is illustrated in Figure 3, which shows the multistep process of creating a nuclear-reactor spent fuel pool [5] from simpler components. The fuel pool is a 24×3 alternating array of reactor assemblies and water assemblies (see Figure 3c). Each of the reactor assemblies is a 15×15 array of pin cells (see Figure 3b), which are composed of an $r = 0.44$

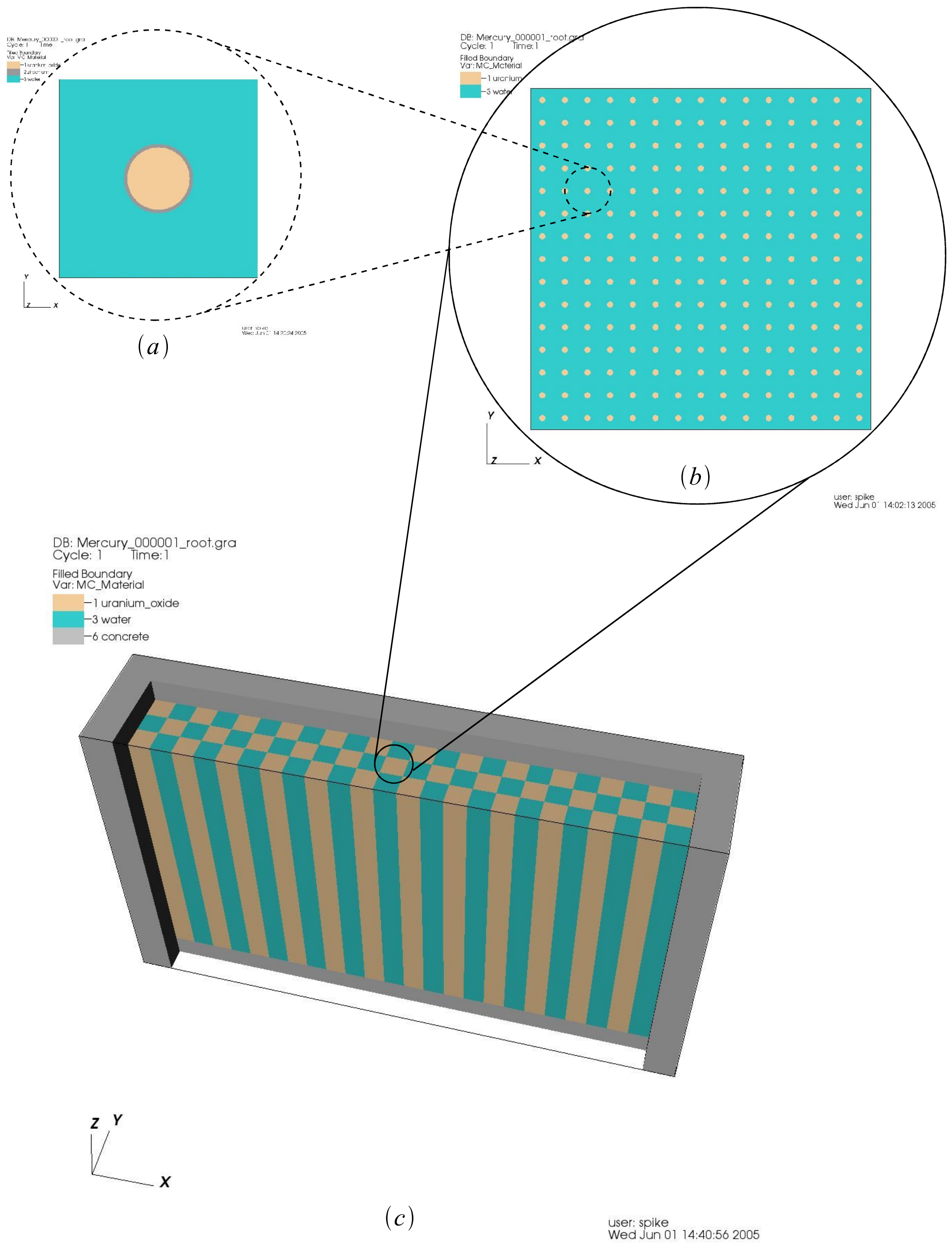


Figure 3. The procedure of generating the geometry for a nuclear-reactor spent fuel pool using templates. The complete system is built from (a) individual pin cells, which are replicated to create (b) a reactor assembly, which are replicated to produce (c) the spent fuel pool.

cm cylindrical uranium-dioxide pin, surrounded by $\delta = 0.05$ cm zirconium cladding that sits within a $\Delta = 1.4$ cm square pitch water channel (see Figure 3a). This array of pin cells is surrounded by a water gap of thickness $\delta = 2.5$ cm. Both the reactor and water assemblies are wrapped by a stainless steel box of thickness $\delta = 0.5$ cm and overall assembly pitch of $\Delta = 27.0$ cm. The collection of assemblies are bounded on three sides by concrete walls and on the remaining three sides by water. The power of this method is evident when one compares the number of template definitions (62) and creation commands (1) that are required to produce all of the cells in the system (24774).

3. NEW PHYSICS CAPABILITIES

Over the past two years a number of new physics capabilities have been added to the code, including an improved neutron thermalization model, a new energy treatment of cross section data and an extension of the scattering model to include bound molecular effects.

3.1 Improved Thermalization Model

A realistic neutron thermalization model has been added to the Monte Carlo All Particle Method (MCAPM) collision library and nuclear data server [6] that is used within MERCURY. This model assumes that the neutrons are elastically scattering off of a background Maxwellian distribution, where the thermal width of the Maxwellian is set by the temperature of the medium. This new model is essentially the same as the method that has been implemented in TART [7]. The previous "floor" thermalization model is also still available for use. That model sets the kinetic energy of a particle to the thermal energy of the cell $E = (3/2)kT$ whenever a collision event would result in a lower secondary particle energy. The elastic scattering model is more realistic and has been set as the default neutron thermalization treatment.

This new model is implemented as an extension of the previously existing collisional kinematics algorithms in MCAPM. When a particle undergoes an elastic scattering event using room-temperature cross sections, MCAPM now checks the ratio of the incident particle energy to the thermal energy of the background medium. If this ratio is $E_{inc}/((3/2)kT) < 1\times10^4$, then the elastic scattering collision is resampled, this time using cross sections that are heated to the correct temperature. Then net effect is that a particle may either elastically up- or down-scatter due to collisions with material that is not at room temperature. This is in stark contrast to the "floor" model, in which particle could only up-scatter.

In order to test this new thermalization model, code-to-code comparisons were made with TART for a reactor pin cell criticality calculation. Problem 1 from [8] was chosen for this purpose. This is a pin cell with pin radius $r = 1.27$ cm in a square water pitch $\Delta = 5.08$ cm on each side. The problem is assumed to be infinite along the axis of the cylindrical pin. The pin is composed of uranium at a density $\rho = 18.8$ g/cm^3 with atomic fractions of $9.902\times10-1$ for ^{238}U and $9.8\times10-3$ for ^{235}U, while the water has atomic fractions of $6.6667\times10-1$ for ^{1}H and $3.3333\times10-1$ for ^{16}O at a density of $\rho = 1.0$ g/cm^3. The heterogeneous nature of this problem makes it ideal to test thermalization of particles in the water moderator region.

Figure 4 shows the production spectrum (top) and absorption spectrum (bottom) obtained from multigroup calculations of the reactor pin cell problem using the MERCURY (red curve) and TART (black curve) codes. Free atom scattering off of hydrogen in the water was assumed for these calculations. Each of these calculations used the same evaluated nuclear data set that was defined on a 616 energy-group structure, with 50 groups per decades spaced equally in lethargy $\ln(E_{max}/E)$ over the range $1.0\times10^{-11} \leq E \leq 20$ MeV. These calculations were each run with 100 million active particle histories. The agreement between the two sets of results is quite good, with only a couple of percent differences over most of the energy range. The k_{eff} computed by the two codes was $k_{eff} = 0.96080\pm0.00013$ and $k_{eff} = 0.96064\pm0.00013$ for MERCURY and TART, respectively. Note that this level of agreement was only possible because the statistical treatment in TART of the unresolved resonance region was disabled, since MERCURY has no

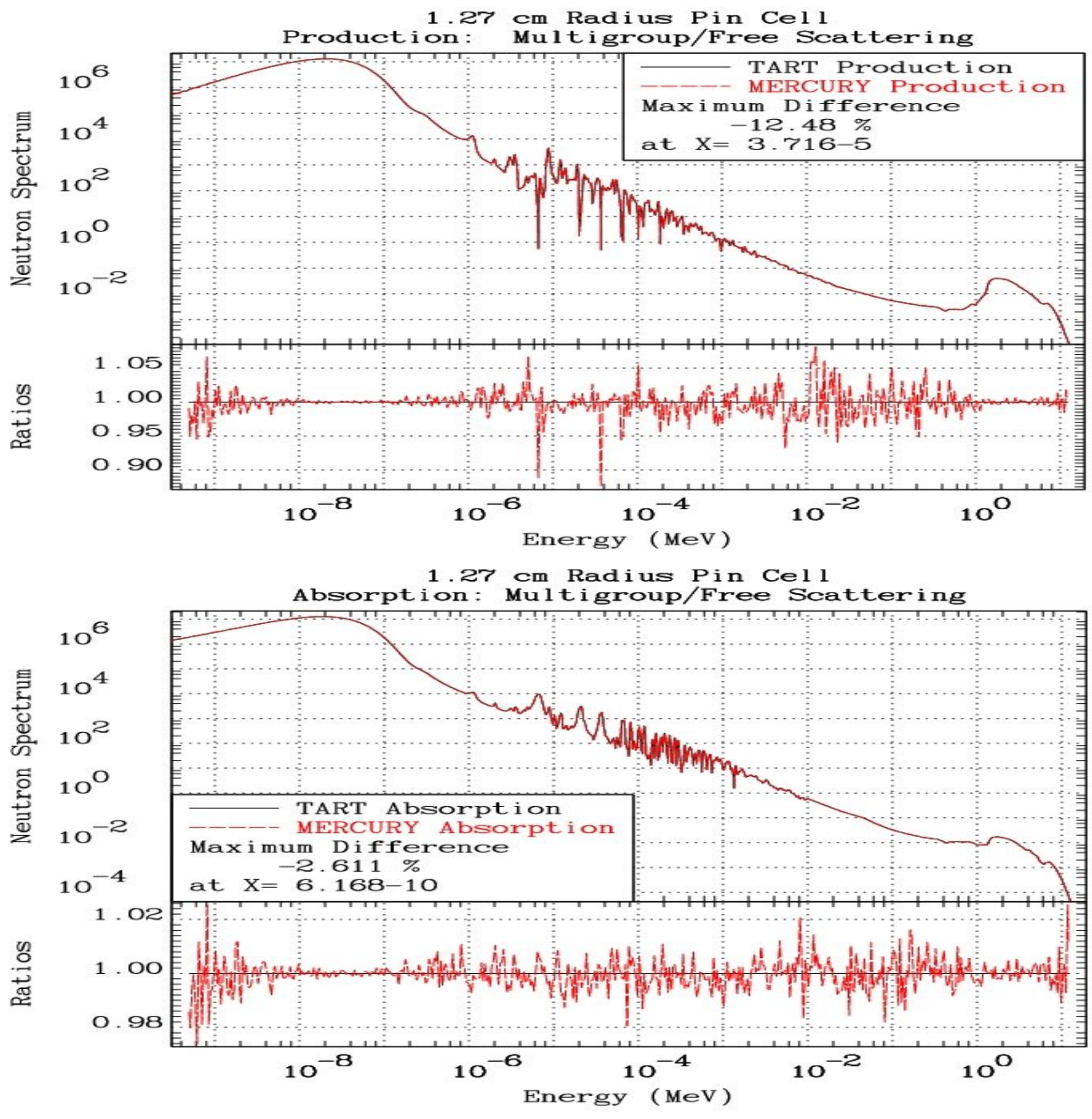


Figure 4. Comparison of the production (top) and absorption (bottom) spectra obtained from multigroup calculations of the pin cell problem using MERCURY (red) and TART (black) using free atom scattering.

such capability. If this feature was not disabled in TART, the difference in the absorption spectrum would be upwards of 15 percent in the energy range $10 \leq E \leq 150$ keV.

3.2 Continuous Energy Cross Sections

The MCAPM library was also extended to provide a continuous energy treatment of cross sections. The pointwise data used for this purpose had always been available within the MCAPM data files, but access routines were not written until recently. If the particle energy lies between two of the energy points at which the cross sections are tabulated, the library linearly interpolates the cross section based upon a linear interpolation of the energy.

This new capability in MERCURY was tested on the same problem that was described in the previous section. The results are shown in Figure 5. Once again the agreement is very good, with a few percent differences over most of the energy range, except in the low-energy, low-probability tail of the production spectrum where the statistical differences can reach 20 percent. The k_{eff} computed by MERCURY was $k_{eff} = 0.99674 \pm 0.00013$, while TART calculated $k_{eff} = 0.99670 \pm 0.00013$ from TART. Note that the difference in k_{eff} of 4×10^{-5} is well within to the statistical uncertainties in each of these 100-million particle history calculations.

3.3 $S(\alpha, \beta)$ Bound Molecular Scattering

Bound molecular scattering was implemented in MCAPM in a straightforward manner. The data describing scattering off of molecular systems can be described with outgoing particle spectra that are energy-angle correlated. Since MCAPM already handled such secondary particle correlations, it was easy to include additional, fictitious isotopes that represented the following molecules into the MCAPM data files:

(1) H in H_2O

(2) H in CH_2

(3) D in D_2O

(4) Be in Metallic Form

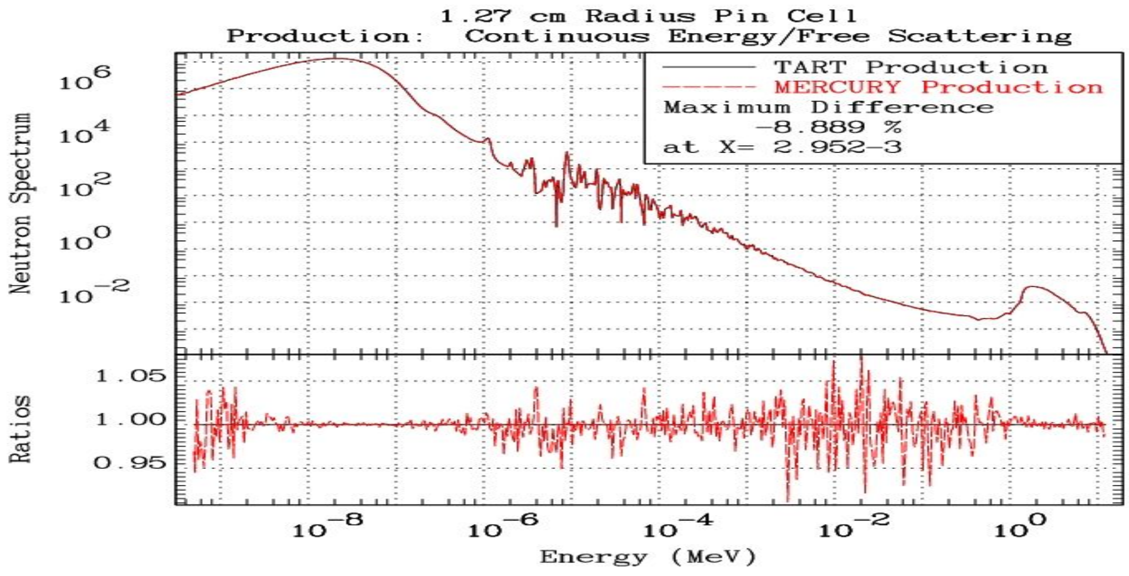
(5) Be in BeO

(6) C in Graphite Form

(7) O in BeO

These isotopes are otherwise identical to their free-atom scattering brethren, except that below a threshold ($E \leq 4$ eV) which is unique for each isotope, the elastic-scattering cross section goes to zero and is replaced by a fictitious inelastic-scattering cross section that represents scattering off of bound molecules. Once again, this is the same treatment that has been previously implemented in TART [9].

Once again, the pin cell calculation that was used above is used to compare the predictions of MERCURY and TART for runs that include $S(\alpha, \beta)$ bound molecular scattering. The k_{eff}



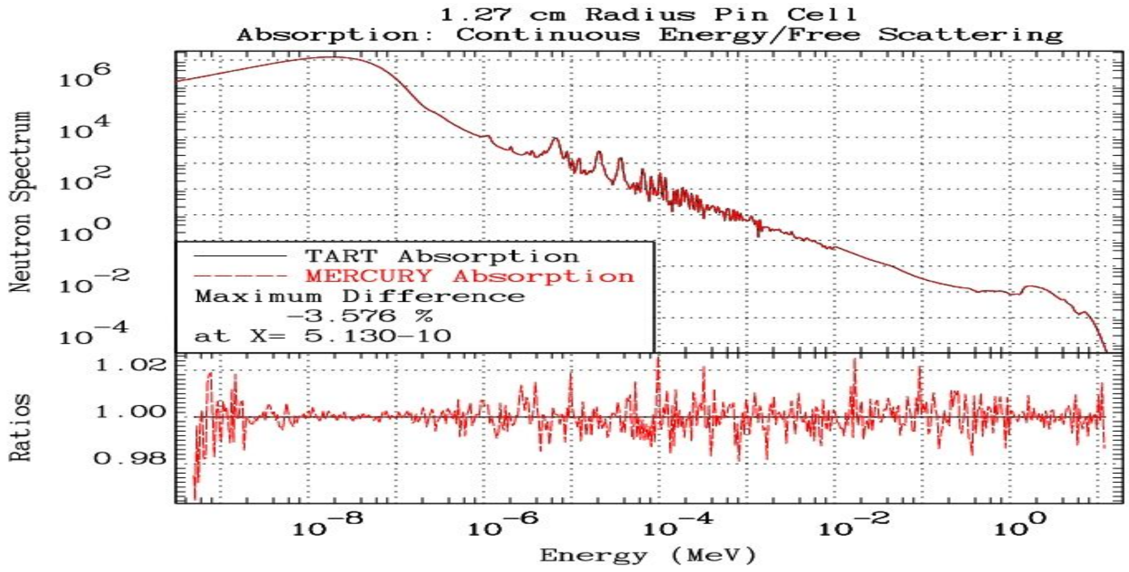
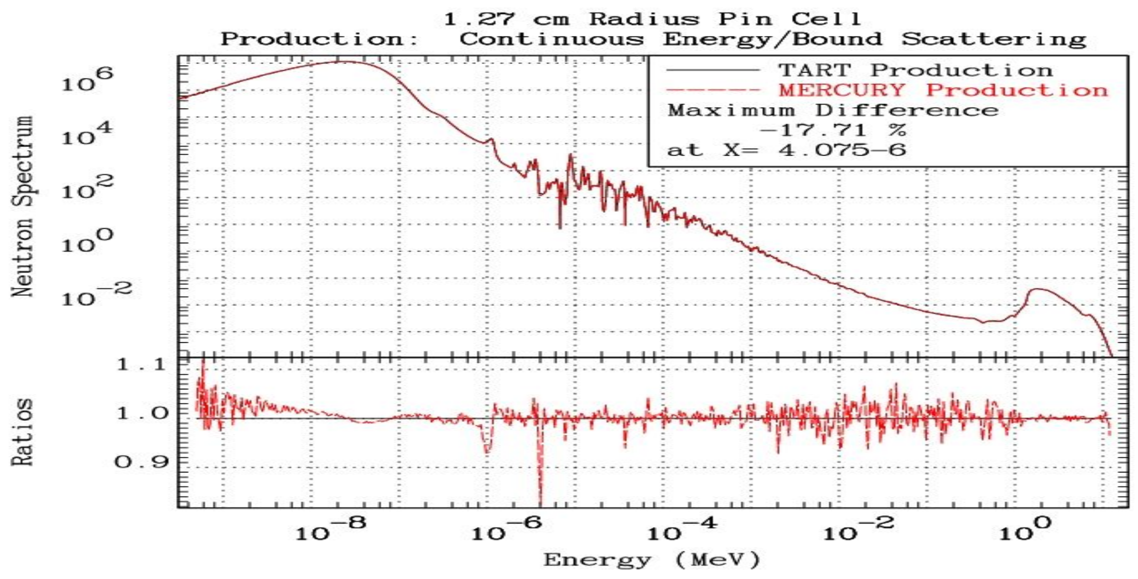


Figure 5. Comparison of the production (top) and absorption (bottom) spectra obtained from continuous energy calculations of the pin cell problem using MERCURY (red) and TART (black) using free atom scattering.

computed by MERCURY was $k_{eff} = 0.94688 \pm 0.00012$ and $k_{eff} = 0.94730 \pm 0.00012$ from TART. The difference in k_{eff} of 4.2×10^{-4} is about 3.5 times the statistical uncertainty of either of these 100-million history calculations. While it is possible that this level of difference is statistically significant, a review of the production and absorption spectra for this problem, which are shown in Figure 6, clearly indicates that there are subtle but significant differences of a few to twenty percent. These differences are occurring only where the $S(\alpha, \beta)$ scattering cross sections are present for $E < 4$ eV. There are indications from other, more rudimentary transport problems that the differences may be due to the method of interpolation within the low- and high-energy equally probable bins from which the secondary energy and scattering angle are chosen. At this point, it is not clear which of the codes is handling the interpolation correctly.



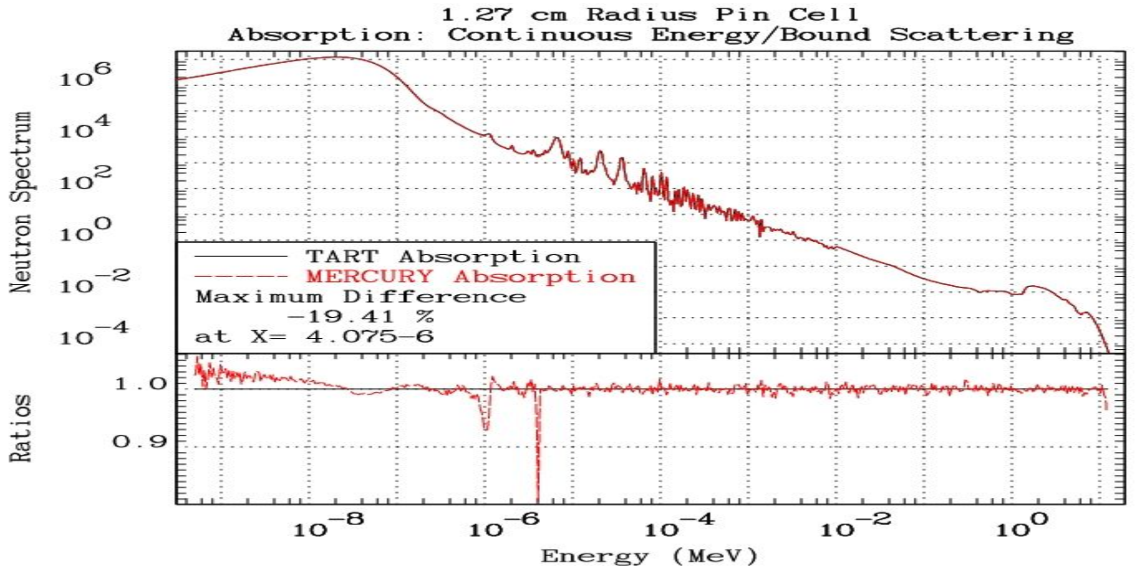


Figure 6. Comparison of the production (top) and absorption (bottom) spectra obtained from continuous energy calculations of the pin cell problem using MERCURY (red) and TART (black) using $S(\alpha,\beta)$ bound molecular scattering.

This discrepancy will continue to be investigated over the coming weeks and will be resolved in the near future.

These results illustrate an important point which must be considered when one attempts to validate a code through code-to-code comparisons. While it is a *necessary* step to compare integral results, such as the k_{eff}, predicted by each code, it is not always *sufficient* to determine that the two codes are actually producing the same results. One should also compare particle spectra produced by the two codes [10]. Comparing the k_{eff} computed for the free-atom and bound-molecular scattering versions of this pin cell, each using continuous energy cross sections, we see that the effect of including bound molecular scattering is to reduce the k_{eff} by about 0.05: a very large change in criticality that is attributed solely to the $S(\alpha, \beta)$ effects. As noted above, the closeness of the k_{eff} computed by the two codes could lead one to conclude that the codes were working correctly, while the spectra of Figure 6 indicates that there are still issues that need to be addressed.

4. RECENT COMPUTER SCIENCE ENHANCEMENTS

Several computer-science related features have been added to the code since 2003. Chief among these are a new input parameter parser which is based upon the XML data-description layer, and a dynamic load balancing capability to improve the performance of parallel calculations.

4.1 An Extensible, XML-Based Input Parameter Parser

The addition of particle tracker for 3-D combinatorial geometries resulted in a significant increase in the number and complexity of MERCURY's input parameters. The input parameter parser that was in the code at the time, while adequate for previous versions of the code, was not easily extended to handle this new level of complexity. A search for a new parser was begun, which led to the use of the XML data-description language and the CYCLOPS system [11]. At its core, CYCLOPS is an XML-based data model that is closely coupled to (1) a text-to-XML input translator, (2) a graphical user interface (GUI) and (3) a set of data-tree query and access routines which use the XERCES parser library [12] for manipulating document object model (DOM) data trees.

Adoption of the CYCLOPS/XERCES parsing system has greatly simplified the addition of new input-parameter blocks into MERCURY. Instead of writing a customized parser for each new block of input parameters (which was in C, with all of its shortcomings in the areas of file manipulation and string/token processing), this new approach allows one to add a new section to the

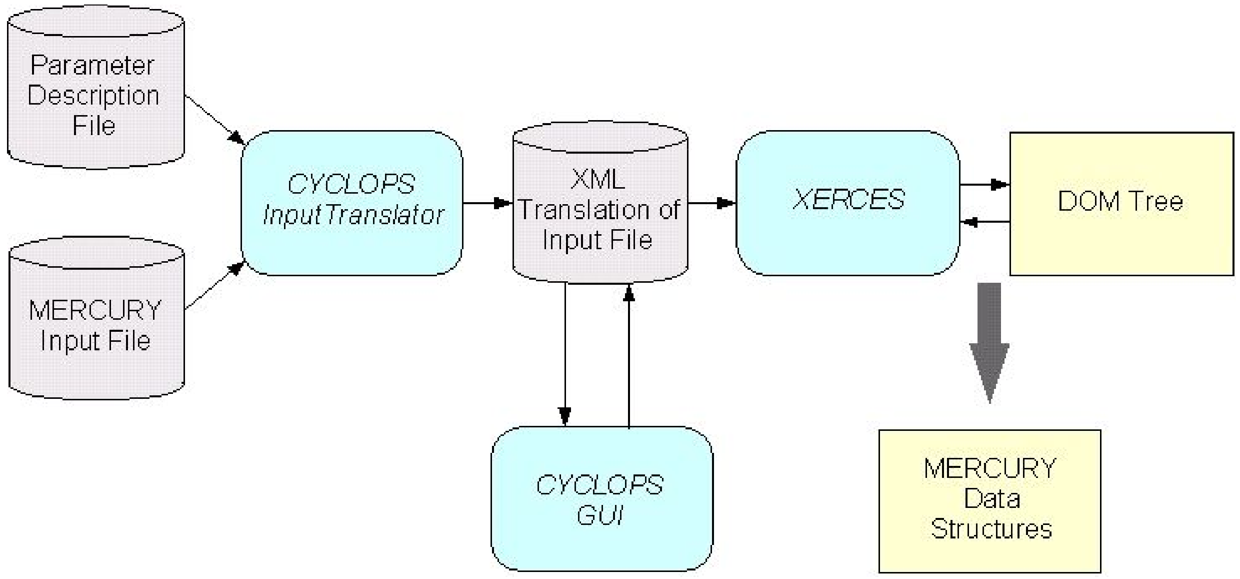


Figure 7. Flow chart of the new XML-based input parameter parser in MERCURY.

data-model description file for each new block of input parameters. All of the file handling features, data querying and access capabilities are modular and used for each of the data blocks.

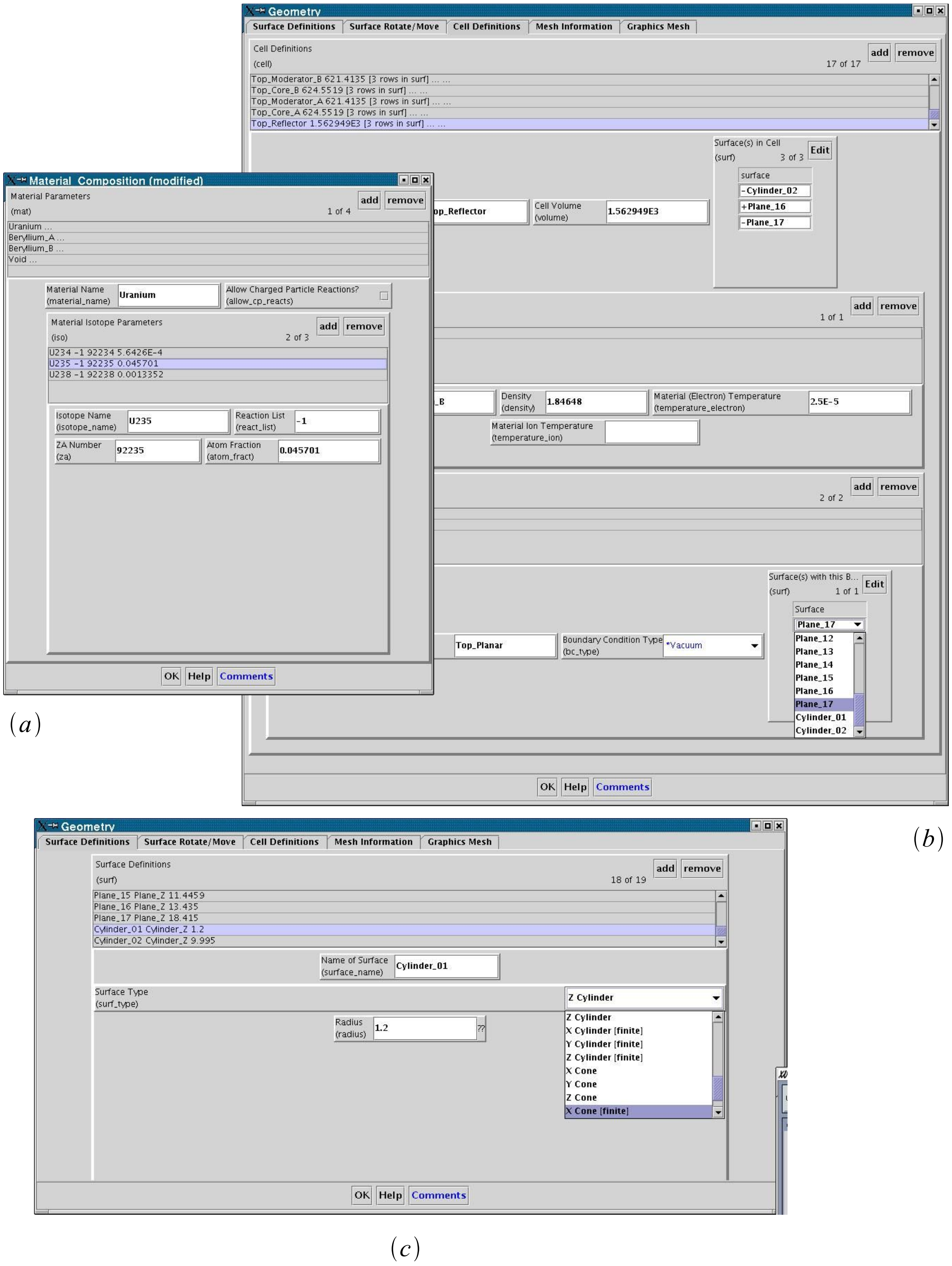
The flow of data through the new parsing system is shown in Figure 7. The CYCLOPS InputTranslator reads both the MERCURY input file and the ParameterDescription file, which contains the definition of the data model for all of MERCURY's input parameters. The InputTranslator produces a simple XML translation of the data in the input file. A DOM tree is created from this simple XML translation. The tree is then interrogated by MERCURY's input routines which use CYCLOPS-wrapped versions of query and access routines from the XERCES library.

One benefit of this process is a rudimentary level of validity checking of the input parameters, which is a feature supported by the CYCLOPS data model as defined in the ParameterDescription file. Another benefit, as shown in the figure, is the CYCLOPS GUI. This graphical interface allows users to either generate new, or modify existing, MERCURY input files. The input-block-based windows, context sensitive help and tooltips can simplify the process of building input files, and can be instructive for the neophyte user of the code.

Several windows from the CYCLOPS GUI are shown in Figure 8. The first is a material-composition window (Figure 8a), which is shown in the process of editing the second of three isotopes in the first of four defined materials. Note that data-entry or check boxes are provided for each required or optional element, along with summary sections that show each of the quantities (isotopes and materials) in this hierarchical data structure. The next two windows show two variants of a geometry window in the process of editing a 3-D combinatorial geometry. The active tab (see the top of the window for all the possible tab choices) in Figure 8b is for cell definition, while in Figure 8c the active tab is for surface definition. Based upon the tab selected, the context of the data entries changes so that only those that are relevant are displayed for the user. As before, summary lists of the defined surfaces, cells and boundary conditions are available for review or editing. Drop-down list boxes are available to select from either predefined options (such as the surface type in Figure 8c) or from a list of user defined entries (such as list of surface in Figure 8b). Clicking the 'Help' button in any of these windows will bring up a web browser which displays the relevant pages from the user's guide [1]. If the user leaves the mouse pointer over any entry box, a tooltip window will appear that contains a brief description of the entry.

4.2 A Dynamic Load Balancing Capability

One of the major design requirements for MERCURY is the ability to run efficiently on a wide variety of computing platforms, from low-end serial desktop systems to the world's largest parallel supercomputers. In order to achieve this, a multiple pronged approach to parallelism has been developed and implemented in the code. The first of these parallel run modes entails a spatial decomposition of the problem geometry into domains, and the assignment individual processors to work on specific domains. This method, known as *domain decomposition*, is a form of spatial parallelism. The second parallel run mode, and the easiest way to parallelize a Monte Carlo transport code, is to store the geometry information redundantly on each of the processors, and assign each processor work on a different set of particles. This method is termed *domain replication*, which is a form of particle parallelism. In many cases, problems are so large that *domain decomposition* alone is not sufficient. For these problems, a combination of both spatial *and* particle parallelism is employed to achieve a scalable parallel solution.



Material Composition (modified)

Material Parameters (mat) 1 of 4

```
Uranium ...
Beryllium_A ...
Beryllium_B ...
Void ...
```

Material Name (material_name): Uranium

Allow Charged Particle Reactions? (allow_cp_reacts)

Material Isotope Parameters (iso) 2 of 3

```
U234 -1 92234 5.6426E-4
U235 -1 92235 0.045701
U238 -1 92238 0.0013352
```

Isotope Name (isotope_name): U235

Reaction List (react_list): -1

ZA Number (za): 92235

Atom Fraction (atom_fract): 0.045701

Geometry — Surface Definitions | Surface Rotate/Move | Cell Definitions | Mesh Information | Graphics Mesh

Cell Definitions (cell) 17 of 17

```
Top_Moderator_B 621.4135 [3 rows in surf] ... ...
Top_Core_B 624.5519 [3 rows in surf] ... ...
Top_Moderator_A 621.4135 [3 rows in surf] ... ...
Top_Core_A 624.5519 [3 rows in surf] ... ...
Top_Reflector 1.562949E3 [3 rows in surf] ... ...
```

op_Reflector

Cell Volume (volume): 1.562949E3

Surface(s) in Cell (surf) 3 of 3

surface: -Cylinder_02, +Plane_16, -Plane_17

1 of 1

_B

Density (density): 1.84648

Material (Electron) Temperature (temperature_electron): 2.5E-5

Material Ion Temperature (temperature_ion):

2 of 2

Top_Planar

Boundary Condition Type (bc_type): *Vacuum

Surface(s) with this B... (surf) 1 of 1

Surface: Plane_17

```
Plane_12
Plane_13
Plane_14
Plane_15
Plane_16
Plane_17
Cylinder_01
Cylinder_02
```

Geometry — Surface Definitions | Surface Rotate/Move | Cell Definitions | Mesh Information | Graphics Mesh

Surface Definitions (surf) 18 of 19

```
Plane_15 Plane_Z 11.4459
Plane_16 Plane_Z 13.435
Plane_17 Plane_Z 18.415
Cylinder_01 Cylinder_Z 1.2
Cylinder_02 Cylinder_Z 9.995
```

Name of Surface (surface_name): Cylinder_01

Surface Type (surf_type): Z Cylinder

```
Z Cylinder
X Cylinder [finite]
Y Cylinder [finite]
Z Cylinder [finite]
X Cone
Y Cone
Z Cone
X Cone [finite]
```

Radius (radius): 1.2

Figure 8. Examples of three input-block windows from the CYCLOPS GUI: (a) the material composition window, (b) the cell-definition tab of the geometry window and (c) the surface-definition tab of the geometry window.

Since particles often migrate in space and time between different regions of a problem, it is a natural consequence of domain decomposition that not all spatial domains will require the same amount of computational work. Hence, the calculation becomes load imbalanced. In many applications, one portion of the calculation (cycle, iteration, etc.) must be completed by all processors before the next phase can commence. If one processor has more work than any of the other processors, the less-loaded processors must wait for the most worked processor to complete its work.

In an attempt to reduce this form of particle-induced load imbalance, a method has been developed which allows the number of processors assigned on a domain to vary dynamically in accordance with the amount of work on that domain [13]. The particles that are located in a given spatial domain are then divided evenly among the number of processors assigned to work on that domain, which is termed the domain's *replication level*. The performance of parallel Monte Carlo transport calculations which use both spatial and particle parallelism is increased by dynamically assigning processors to the most worked domains. Since he particle work load varies over the course of the simulation, this algorithm determines each cycle if dynamic load balancing would speed up the calculation. If load balancing is required, a small number of particle communications are initiated in order to achieve load balance. This method has demonstrated a decrease in the parallel run time by more than a factor of two for certain criticality and source calculations [14].

5. SUMMARY AND FUTURE DIRECTIONS

This paper has provided and update on the development and validation of the MERCURY Monte Carlo particle transport code. Significant progress has been made during the course of the past two years in several areas, including the development of algorithms for tracking particles (a 3-D combinatorial geometry (CG) tracker, as well as trackers for 1-D spherical/radial and 2-D arbitrary quadrilateral meshes; templates for the definition of complex hierarchical geometries with repeated structures), the addition of new physics capabilities (an improved thermalization model, continuous energy representation of cross sections and $S(\alpha,\beta)$ bound-molecular scattering effects) and a number of important computer science enhancements (an extensible XML-based input parameter parser and a method of dynamically balancing the load during parallel calculations).

Several new capabilities are expected to be added to the code over the next one to two years. The most important set of enhancements are an extension of the current tally and source capabilities which will allow the user to tally into, or sample from, an *n*-dimensional distribution, where the individual dimensions can be time, energy, angle(s), a 2-D or 3-D Cartesian mesh, CG surfaces, CG cells, etc. The core tally module will also be used in a post-processing tool named CALORIS, which is designed to tally particles that are written to disk during a prior MERCURY calculation. In addition, CALORIS may also be used to filter particles according to a set of criteria in order to develop a source for a subsequent MERCURY calculation. The next set of algorithms to be developed relate to variance reduction. Geometry-based population control, weight windows, detector biasing, collisional survival biasing and an exponential transform method will be added during the next two years.

Additional areas of planned code development include completion of the energy deposition and isotopic burnup capabilities (currently undergoing testing), extension of the CG cell definition syntax to include other Boolean operators besides `AND', the ability to embed meshes within combinatorial geometries, particle trackers for 2-D $r - z$ cylindrical and 3-D Cartesian adaptive mesh refinement (AMR) meshes, implementation of a non-adjoint method for calculating the probability of initiation, and implementation of the VISIT visualization tool within MERCURY. Planned enhancements of the MCAPM nuclear data and collision library include an unresolved-resonance-region treatment, a multi band statistical resonance treatment and the addition of delayed neutrons.

ACKNOWLEDGMENTS

This work was performed under the auspices of the U.S. Department of Energy by University of California, Lawrence Livermore National Laboratory under Contract Number W-7405-Eng-48.

REFERENCES

1. R. J. Procassini and J. M. Taylor, *MERCURY User Guide (Version b.8)*, Lawrence Livermore National Laboratory, Report UCRL-TM-204296, Revision 1 (2005).

2. D. E. Cullen, *TART 2002: A Coupled, Neutron-Photon 3-D, Combinatorial Geometry Time Dependent Monte Carlo Transport Code*, Lawrence Livermore National Laboratory, Report UCRL-ID-126455, Revision 4 (2002).

3. R. Buck, E. Lent, T. Wilcox and S. Hadjimarkos, *COG User's Manual: A Multiparticle Monte Carlo Transport Code (Fifth Edition)*, Lawrence Livermore National Laboratory, Internal Report (2002).

4. R. J. Procassini, J. M. Taylor, I. R. Corey and J. D. Rogers, "Design, Implementation and Testing of MERCURY: A Parallel Monte Carlo Transport Code", *2003 Topical Meeting in Mathematics and Computations*, Gatlinburg, TN, 7 - 10 April 2003, American Nuclear Society (2003).

5. N. R. Smith, et al., "OECD/NEA Source Convergence Benchmark 1: Checkerboard Storage of Assemblies", http://www.nea.fr/html/science/wpncs/convergence/specifications/b1-checker-board.pdf, Organization for Economic Cooperation and Development, Nuclear Energy Agency (2001).

6. P. S. Brantley, C. A Hagmann and J. A. Rathkopf, *MCAPM-C Generator and Collision Routine Documentation (Revision 1.2)*, Lawrence Livermore National Laboratory, Report UCRL-MA-141957 (2003).

7. D. E. Cullen, *THERMAL: A Routine Designed to Calculate Neutron Thermal Scattering (Revision 1)*, Lawrence Livermore National Laboratory, Report UCRL-ID-120560 (1995).

8. D. E. Cullen, R. N. Blomquist, C. Dean, D. P. Heinrichs, M. A. Kalugin, M. Lee, Y-K Lee, R. MacFarlane, Y. Nagaya and A. Trkov, *How Accurately Can We Calculate Thermal Systems?*, Lawrence Livermore National Laboratory, Report UCRL-TR-203892 (2004).

9. D. E. Cullen, L. F. Hansen. E. M. Lent and E. F. Plechaty, *Thermal Law Scattering Data: Implementation and Testing Using the Monte Carlo Neutron Transport Codes COG, MCNP and TART*, Lawrence Livermore National Laboratory, Report UCRL-ID-153656 (2003).

10. R. J. Procassini, D. E. Cullen, G. M. Greenman and C. A. Hagmann, "Verification and Validation of MERCURY: A Modern Monte Carlo Particle Transport Code", *Monte Carlo 2005: Topical Meeting in Monte Carlo*, Chattanooga, TN, 17 - 21 April 2005, American Nuclear Society (2005).

11. N. H. Samuelson, *Cyclops User's Manual (Version 2.0)*, Lawrence Livermore National Laboratory, Internal Report (2003).

12. The XERCES Team, "XERCES C++ Parser", http://xml.apache.org/xerces-c/, The Apache XML Project, The Apache Software Foundation (2004).

13. M. J. O'Brien, J. M. Taylor and R. J. Procassini, "Dynamic Load Balancing of Parallel Monte Carlo Transport Calculations", *Monte Carlo 2005: Topical Meeting in Monte Carlo*, Chattanooga, TN, 17 - 21 April 2005, American Nuclear Society (2005).

14. R. J. Procassini, M. J. O'Brien and J. M. Taylor, "Load Balancing of Parallel Monte Carlo Transport Calculations", *2005 Topical Meeting in Mathematics & Computations*, Avignon, France, 12 - 15 September 2005, American Nuclear Society (2005).